

## Adaptive Mobile Applications (Continuous Connectivity)

Anthony D. Joseph  
CS 294-1

Lecture 15  
December 6, 2000

## Outline

- Projects
  - Poster session: Dec 11, 2-4 pm
  - Papers due Dec 12 by 5 pm (10-12 pgs)
- Today: Continuous Connectivity
  - More wireless devices and networks
    - Fewer islands of disconnection
    - Better network coverage with overlay networks
  - But, quality of coverage varies
    - Bandwidth, latency, jitter, cost, ...
  - Need for applications (or networks) that can adapt to these changes

December 6, 2000

CS 294-1 Lecture #15

2

## Papers

- Agile Application-Aware Adaptation for Mobility [Noble et al '97]
- Proxy-based approaches
  - Adapting to Network and Client Variation via On-Demand, Dynamic Distillation [Fox et al '96]
  - Adapting to network and client variation using active proxies: lessons and perspectives [Fox et al '98]
- Active network vision and reality: lessons from a capsule-based system [Wetherall '99]

December 6, 2000

CS 294-1 Lecture #15

3

## Agile Application-Aware Adaptation for Mobility [Noble et al '97]

- Three key features of application adaptation for mobility
- Fidelity
  - Quality of content
- Concurrency
  - Resource sharing/management
- Agility
  - Speed/flexibility of adaptation

December 6, 2000

CS 294-1 Lecture #15

4

## Fidelity

- Degree to which data at a client matches the copy at the server
- Fidelity has many dimensions
  - Depends on the type of data in question (video data has 2 or more dimensions)
- The dimensions of fidelity are natural axes of adaptation for mobility
- Real time apps (video conferencing) are excluded from adaptation

December 6, 2000

CS 294-1 Lecture #15

5

## Concurrency

- Ability to execute multiple independent applications on a mobile client is vital
- Operating systems must now manage a broader range of resources such as network bandwidth, disk cache space and battery power

December 6, 2000

CS 294-1 Lecture #15

6

## Agility

- Speed and accuracy with which the system detects and responds to changes in resource availability
- Sensitivity to different resources. (network bandwidth, battery power)
- Different origins of changes in resource availability
- Variation in the *supply* of a resource due to mobility
- Variation by changed *demand* for it by concurrent applications

December 6, 2000

CS 294-1 Lecture #15

7

## Application-Aware Adaptation Process

- The system monitors resource levels, and notifies applications of relevant changes
- Each application decides how to best adapt when notified

December 6, 2000

CS 294-1 Lecture #15

8

## Supporting Application Diversity and Concurrency

- Diversity: Allow applications to determine the mapping of resource levels to fidelity levels
- Concurrency: Allow the system to retain control of resource monitoring
- Adaptation: Entirely handled by applications

December 6, 2000

CS 294-1 Lecture #15

9

## Where Adaptation Occurs

- In Coda, adaptation is entirely managed by the system
- In Odyssey, system code treats data generically
- Individual applications are responsible for differential handling of data types

December 6, 2000

CS 294-1 Lecture #15

10

## Type Awareness

- Wide disparity in the properties of various data types
  - Some form of type awareness should be incorporated into the system for efficient resource usage
- Impossible to optimize without some system level knowledge of type
  - Image data may be highly compressible using one algorithm but not another
  - Video data can be efficiently shipped using a streaming protocol that drops rather than retransmits lost data

December 6, 2000

CS 294-1 Lecture #15

11

## Type Awareness in Odyssey

- Odyssey incorporates type-awareness via specialized code called wardens
- A warden encapsulates the system-level support at a client
  - To support a new data type, an appropriate warden has to be written and incorporated into Odyssey at each client
  - A type-independent component called viceroy is responsible for centralized resource management

December 6, 2000

CS 294-1 Lecture #15

12

## Data vs. Action-Centric Adaptation

- *Data-Centric Adaptation*
  - Between *viceroy* and *wardens*
  - It defines the fidelity levels for each data type and factors them into resource management
- *Action-Centric*
  - Between applications and Odyssey
  - It provides applications with control over the selection of fidelity levels supported by the wardens

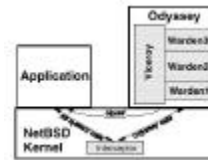
December 6, 2000

CS 294-1 Lecture #15

13

## Design and Implementation

- Odyssey is built upon NetBSD file system calls
- Viceroy and wardens are implemented in user space rather than in kernel
  - Follows MACH microkernel model



December 6, 2000

CS 294-1 Lecture #15

14

## Design and Implementation

- Communication between the viceroy and wardens is through procedure calls and shared data structures
- The wardens are entirely responsible for communicating with servers and caching data
- Applications never contact servers directly

December 6, 2000

CS 294-1 Lecture #15

15

## Expressing Resource Expectations

- Applications communicate resource expectations to Odyssey using the **request** system call
  - request** ( in *path*, in *resource descriptor*, out *request-id*)
  - cancel** (in *request-id*)
- The call takes a resource descriptor identifying a resource and a *window of tolerance*

December 6, 2000

CS 294-1 Lecture #15

16

## Expressing Resource Expectations

- Viceroy registers the request returning a unique id if the availability of the resource lies within the window of tolerance
- The id is used in notifying the app that the resource has left the required bounds via **cancel**
- If the resource is outside the bounds, the current resource level is returned

December 6, 2000

CS 294-1 Lecture #15

17

## Notifying Applications

- The viceroy generates an *upcall* to the application, when it discovers that the availability of a resource is outside the window
- The application adjusts the fidelity according to its policy
- It then issues another **request** call to register a revised window of tolerance to the new fidelity

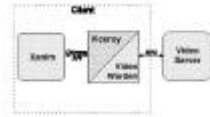
December 6, 2000

CS 294-1 Lecture #15

18

## Example Applications: Video Player

- *Xanim* video animation software
- Three levels of fidelity : JPEG(50), JPEG(99), and black-and-white frames



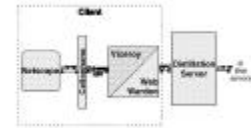
December 6, 2000

CS 294-1 Lecture #15

19

## Example Apps: Web Browser

- Netscape requests are redirected to a client module called *cellophane*
- Cellophane selects fidelity levels and transforms HTTP requests into file operations on Odyssey Web objects
- A distillation server provides multiple levels of fidelity for images
- The warden provides a mechanism to set the fidelity level



December 6, 2000

CS 294-1 Lecture #15

20

## Evaluation

- How agile is Odyssey in the face of changing network bandwidth?
- How beneficial is it for applications to exploit the adaptation that Odyssey offers?

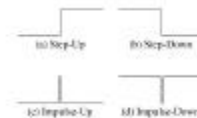
December 6, 2000

CS 294-1 Lecture #15

21

## Experimental Methodology

- Reference Waveforms for Agility Experiments
- Generating Discontinuous Waveforms



December 6, 2000

CS 294-1 Lecture #15

22

## Experimental Methodology

- The observations are logged by a user-level RPC mechanism which is implemented on UDP
- Log entries: round trip entries for small exchanges, throughput entries that arise from bulk transfers
- Throughput entry: time for a receiver to request and receive a window's worth of D data
- Bandwidth estimate from round trip and throughput entries
- The viceroy collects information from all logs to estimate the total bandwidth available to the client

December 6, 2000

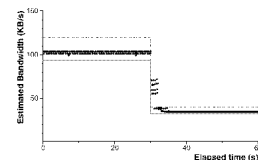
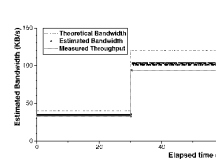
CS 294-1 Lecture #15

23

## Varying Supply

Step-up waveform

Step-down waveform



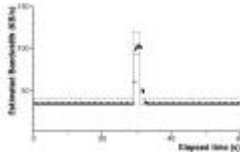
December 6, 2000

CS 294-1 Lecture #15

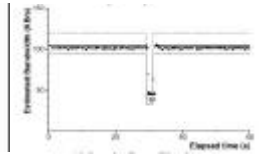
24

## Varying Supply

Impulse-up



Impulse-down



December 6, 2000

CS 294-1 Lecture #15

25

## How Beneficial is Adaptation?

Video Player performance and fidelity

| Waveform     | BW Fidelity = 0.01 |          | JPEG(S) Fidelity = 0.5 |          | JPEG(S) Fidelity = 1.0 |          | Odyssey |          |      |        |
|--------------|--------------------|----------|------------------------|----------|------------------------|----------|---------|----------|------|--------|
|              | Drops              | Fidelity | Drops                  | Fidelity | Drops                  | Fidelity | Drops   | Fidelity |      |        |
| Step-Up      | 0                  | 0.00     | 3                      | (1.0)    | 169                    | (5.0)    | 7       | (2.2)    | 0.73 | (0.01) |
| Step-Down    | 0                  | 0.00     | 5                      | (1.3)    | 169                    | (5.0)    | 25      | (8.8)    | 0.76 | (0.01) |
| Impulse-Up   | 0                  | 0.00     | 3                      | (0.7)    | 325                    | (4.3)    | 23      | (7.4)    | 0.50 | (0.01) |
| Impulse-Down | 0                  | 0.00     | 0                      | 0.00     | 12                     | (5.7)    | 14      | (6.5)    | 0.98 | (0.01) |

December 6, 2000

CS 294-1 Lecture #15

26

## Experimental Results

Web Browser

| Waveform     | JPEG(S) Fidelity = 0.05 |          | JPEG(S) Fidelity = 0.25 |          | JPEG(S) Fidelity = 0.5 |          | FIT Quality Fidelity = 1.0 |          | Odyssey  |          |      |        |
|--------------|-------------------------|----------|-------------------------|----------|------------------------|----------|----------------------------|----------|----------|----------|------|--------|
|              | Time (s)                | Fidelity | Time (s)                | Fidelity | Time (s)               | Fidelity | Time (s)                   | Fidelity | Time (s) | Fidelity |      |        |
| Entered      | ---                     | ---      | ---                     | ---      | ---                    | ---      | ---                        | ---      | ---      | ---      |      |        |
| Step-Up      | 0.25                    | (0.00)   | 0.50                    | (0.01)   | 0.29                   | (0.00)   | 0.46                       | (0.01)   | 0.35     | (0.05)   | 0.38 | (0.00) |
| Step-Down    | 0.25                    | (0.00)   | 0.10                    | (0.01)   | 0.29                   | (0.00)   | 0.46                       | (0.00)   | 0.35     | (0.00)   | 0.77 | (0.00) |
| Impulse-Up   | 0.27                    | (0.00)   | 0.33                    | (0.01)   | 0.36                   | (0.00)   | 0.71                       | (0.00)   | 0.42     | (0.00)   | 0.63 | (0.00) |
| Impulse-Down | 0.24                    | (0.00)   | 0.27                    | (0.02)   | 0.29                   | (0.00)   | 0.34                       | (0.01)   | 0.36     | (0.00)   | 0.69 | (0.01) |

December 6, 2000

CS 294-1 Lecture #15

27

5-minute Break

December 6, 2000

CS 294-1 Lecture #15

28

## Adapting to Network and Client Variation via On-Demand Dynamic Distillation and Application Partitioning [Fox '96/'98]

• Motivation

- Mantra: *Access Is the Killer App*
  - PDA's, smart phones, laptops, 2-way pagers...
- Client variation is problematic for servers

| Property  | Desktop + Fast LAN | PDA + wide-area wireless |
|-----------|--------------------|--------------------------|
| Bandwidth | 10-100 Mb/s        | 4800-19.2 Kb/s           |
| Display   | 1280x1024x16       | 320x240x2                |
| CPU       | 133 MHz Pentium    | 20 MHz ARM               |
| Memory    | 10's of MB         | 2-4 MB                   |

December 6, 2000

CS 294-1 Lecture #15

29

## Goals

- Reduce *end-to-end latency* when link is slow
- *Meaningful* presentation for range of clients
- Adapt to network changes at application level
- New abilities for limited clients

December 6, 2000

CS 294-1 Lecture #15

30

## Design Principles

1. Datatype-Specific Distillation & Refinement
2. Distillation On Demand
3. Push computing into infrastructure
4. Application Partitioning

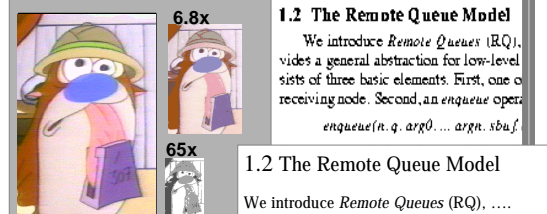
December 6, 2000

CS 294-1 Lecture #15

31

## Datatype-Specific Distillation

- Lossy compression that preserves semantic content
- Tailor content for each client



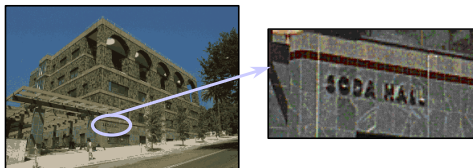
December 6, 2000

CS 294-1 Lecture #15

32

## Refinement

- Retrieve part of distilled object at higher quality



**Distilled image  
(by 60X)**

**Zoom in to original  
resolution**

December 6, 2000

CS 294-1 Lecture #15

33

## Distillation On Demand

- Fine-tune content for heterogeneous devices
  - Target data formats
- User preference profile drives distillation
- Dynamic adaptation as bandwidth varies
  - Dynamic+caching may be more efficient than precomputing

December 6, 2000

CS 294-1 Lecture #15

34

## Computing in the Infrastructure

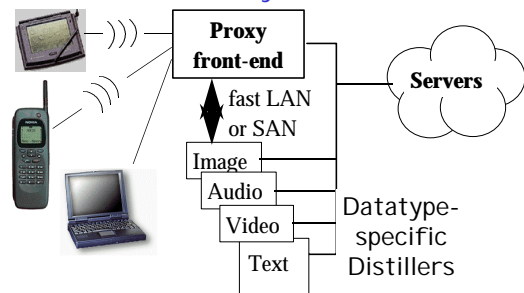
- Legacy server compatibility
- Partition functionality for "small" clients
  - Amortize resources across many clients
- Can combine with distributed caching
- Place inside "boundary of connectivity"

December 6, 2000

CS 294-1 Lecture #15

35

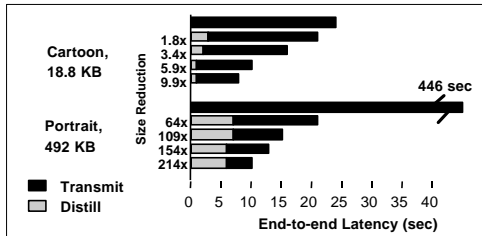
## Distillation Proxy Architecture



December 6, 2000

36

## End-to-End Latency for Images



- Distillation introduces latency into critical path
- But, reduces transmission time

December 6, 2000

CS 294-1 Lecture #15

37

## End-to-End Latency Reduction

- End-to-end latency reduced by 3x-10x
  - Images: 3x-100x, 10x common
  - Rich text: 5x-7x (PostScript to HTML)
- How important is datatype-specific distillation?
  - *gzip* images: < 3%, and takes too long

*Datatype-specific distillation reduces end-to-end latency, often by 10x*

December 6, 2000

CS 294-1 Lecture #15

38

## Extending Client Capabilities

- Watch MBONE Video at home
  - Real-time transcoding of 1-2 Mbps source stream
  - Tune frame rate, frame size, color depth
  - 28.8 Kbs: 1-4 f/s (160 x 120, 8 bit color)
  - ISDN: 10-15 f/s
- Read PostScript on your smart phone
  - PostScript to HTML distiller: 5x-7x reduction

*Move complexity from client to infrastructure*

December 6, 2000

CS 294-1 Lecture #15

39

## Application Partitioning

- *Problem:* "Porting" to small devices is out of the question
  - Dissimilar development environments
  - Lack of Unix or Windows tools and semantics
  - Limited memory, CPU, filesystem
  - Bizarre programming model
- *Goal:* Write as little code as possible.
- What about Windows CE? and Java?

December 6, 2000

CS 294-1 Lecture #15

40

## Application Partitioning By Proxy

*Partitioning goals:*

- Re-use existing desktop-client code
- Exploit client's "natural" abilities when possible
  - GUI, pen input, built-in applications, etc.

December 6, 2000

CS 294-1 Lecture #15

41

## Thin Clients: *Top Gun Wingman*

- World's only graphical browser for PalmPilot handheld device
- >5000 users worldwide
- Richest feature set of any Pilot browser, many uniquely enabled by proxy
- #9 out of 100 in "People's Choice" awards



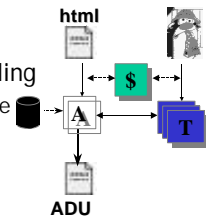
December 6, 2000

CS 294-1 Lecture #15

42

## Top Gun Wingman as a Thin Client

- Intermediate-form page layout
- Image scaling & transcoding
  - Controlled by layout engine
- Device-specific ADU marshalling
  - Including client versioning
  - Originals and device-specific pages cached



December 6, 2000

CS 294-1 Lecture #15

43

## Partitioning Summary

- Exploit resources at infrastructure proxies
  - connectivity
  - Unix/desktop programming idioms, environment
  - allows *aggressive* partitioning of client code
- Reuse existing desktop-client code
  - a lot less to debug!
- Exploit client's "natural" capabilities

*Move complexity away from "small" clients*

December 6, 2000

CS 294-1 Lecture #15

44

## Conclusions

- Datatype-specific distillation:
  - Substantially reduces end-to-end latency
  - Refinement allows bandwidth management
- On-demand distillation tunes content for devices, users, and networks
- Computation in the infrastructure:
  - Simplifies clients by enabling aggressive partitioning
  - Enables new client capabilities

*Next step: scalable uniform proxy architecture*

December 6, 2000

CS 294-1 Lecture #15

45

## Evolution: TACC Servers

- TACC Service
  - *Transformation, Aggregation, and Caching* of existing content
  - *Customization* tunes behavior to each user
- Scalable TACC architecture meets the challenges of infrastructural services:
  - Scalability and high availability
  - Low incremental operating cost

*Rapid prototyping and deployment of new services*

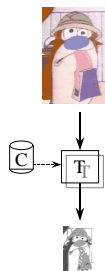
December 6, 2000

CS 294-1 Lecture #15

46

## TranSend, a Prototype TACC Server

- Lossy compression of Web images
  - Downloads 3-7x faster
  - Datatype-specific transformers
- Customize quality/speed tradeoff
  - Everything done on the fly
- TACC unifies large classes of existing services
  - Search, directory, cache
  - Filtering, dynamic content generation,



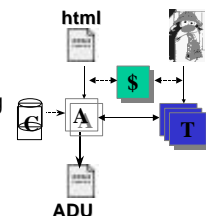
December 6, 2000

CS 294-1 Lecture #15

47

## Wingman as a TACC Application

- PalmPilot web browser
- Intermediate-form page layout
- Image scaling & transcoding
  - Controlled by layout engine
- Device-specific ADU marshalling
  - Including client versioning
  - Originals and device-specific pages cached



December 6, 2000

CS 294-1 Lecture #15

48

## Example: Bay Area Culture Page

- Aggregate and collate static content
  - uses dates as "landmarks" (layout independence)
  - configurable start, end dates, and content sites

### Events between 10 Feb 1997 and 25 Dec 1997

- 15 Feb 1997 - 7 p.m., Zellerbach Hall Shenyang Peking Opera Company at [Café Performance](#)
- 16 Feb 1997 - 2 p.m. and 8 p.m., at [Café Performance](#)
- 20 Feb 1997 - 7 p.m., Zellerbach Hall Baaba Maal at [Café Performance](#)
- 23 Feb 1997 - 8 p.m., Zellerbach Hall Vellinger String Quartet at [Café Performance](#)
- 25 Feb 1997 - 7 p.m., Hertz Hall The Gospel at Colonus Tuesday at [Café Performance](#)
- 1 Mar 1997 - 20, 8 p.m., Zellerbach Hall March The Gospel at Colonus at [Café Performance](#)
- 2 Mar 1997 - 2 p.m. and 8 p.m., at [Café Performance](#)
- 6 Mar 1997 - 3 p.m., Zellerbach Hall Mark Morris Dance Group & Yo-Yo Ma Thursday & at [Café Performance](#)

December 6, 2000

CS 294-1 Lecture #15

49

## Cluster-Based Scalable Internet Services

- TranSend & Wingman* deployed on UCB NOW
  - Nearly continuous uptime since Apr. 1997
- Automatic scaling, availability, load balancing
  - Simple composition-based programming model
  - Semantics allow cluster-friendly implementation
  - Workers shielded from most failure modes
- Focus on the content of the service*
  - New infrastructural services easier than ever to deploy

December 6, 2000

CS 294-1 Lecture #15

50

## Active Network Vision and Reality: Lessons from a Capsule-based System [Wetherall'99]

- Motivation for active networks
- ANTS, active network system
  - Most widespread use of an active nets system
- Lessons learned from ANTS

December 6, 2000

CS 294-1 Lecture #15

51

## Active Networks — Motivation

- Network change is excruciatingly slow
  - Widespread consensus, manual deployment
- Consider:
  - Congestion (RED '93); More Addresses (IPv6 '91)
  - Security (IPSEC '93); Multi-point (IP multicast '90)
- Contrast:
  - Web ('93); Quake ('96); Instant Messaging ('98)

Worth fixing? Your call.

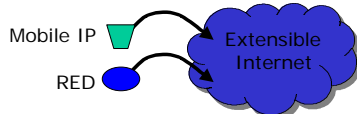
December 6, 2000

CS 294-1 Lecture #15

52

## Active Networks — Approach

- Apply mobile code!
  - Ease deployment; accelerate pace of innovation

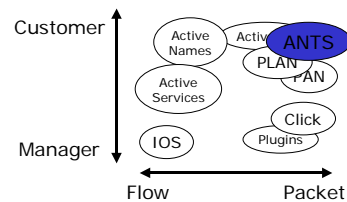


December 6, 2000

CS 294-1 Lecture #15

53

## Kinds of Active Networks



ANTS explores an aggressive vision

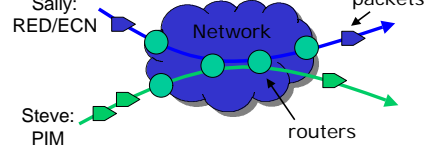
December 6, 2000

CS 294-1 Lecture #15

54

## ANTS, Starting at 10,000 ft

Goal: Let each user control their packets



How do we build this?

December 6, 2000

CS 294-1 Lecture #15

55

## Active Nodes Host New Services

- Nodes provide API for new service code
  - Soft-storage, routing, environment queries, packet manipulation
- Nodes run service code safely
  - Protect state at node; enforce packet invariants
- Nodes manage local resources
  - Bound code runtimes and other resources

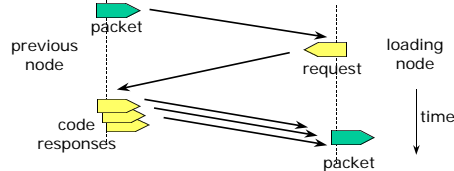
December 6, 2000

CS 294-1 Lecture #15

56

## Code Follows Packets As Needed

- Tie code distribution to packet forwarding
  - End-systems pre-load code, active nodes load hop-by-hop as needed and then cache

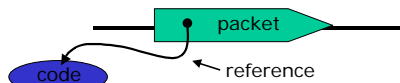


December 6, 2000

CS 294-1 Lecture #15

57

## Packets Refer To Code They Need



- Reference is based on fingerprint
  - Efficient: MD5 is 128 bits, quick to compute
  - Prevents code spoofing: verify without trust
  - No need for standards body: distributed naming

December 6, 2000

CS 294-1 Lecture #15

58

## Implementation — ANTS Toolkit

- User-level reference platform
  - ~10,000 lines, 100% Java, UDP overlay
  - Nodes build on Java protection
- Publicly released for two years
  - <http://www.cs.washington.edu/homes/djw/ants/>
  - Used at MIT, Utah, TIS, TASC, SRI, UIUC, UCLA, ...

December 6, 2000

CS 294-1 Lecture #15

59

## Lessons Learned

Interested in three dimensions:

- Applications
- Performance
- Security and Resource Management

December 6, 2000

CS 294-1 Lecture #15

60

## Application Lessons

- Q: Can we deploy important services?
- Chosen approach lets users select independently
  - Programs constrained by active nodes
- A: Yes
- Well-suited to experimenting w/ protocol variations (rather than computation pushed into network)
  - Exceptions: enforcing policy at a point (firewalls) and resource control (guaranteed service)

December 6, 2000

CS 294-1 Lecture #15

61

## Services That Have Been Written

- Auctions
  - Web cache diversion routing
  - "TCP-SYN" filtering
  - Reliable Multicast support
  - Multicast (single source, PIM)
  - Mobility
  - Path MTU discovery
- novel  
↑  
↓  
established

December 6, 2000

CS 294-1 Lecture #15

62

## 3b. Performance Lessons

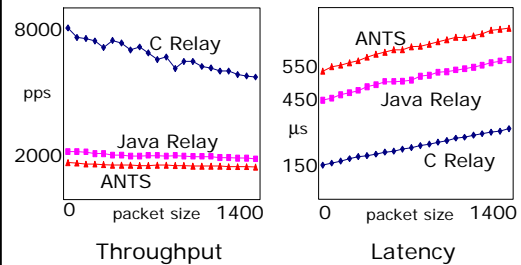
- Q: Is performance a show-stopper?
- Model is more expensive than IP
- A: No
- Very few extra steps over IP in common case and extra steps (demux, safe eval) known to run fast
  - Balance available vs. required computation
    - e.g. 1GHz, 1Gbps, 1000b, 100% → 1000 cycles

December 6, 2000

CS 294-1 Lecture #15

63

## Measured Performance



December 6, 2000

CS 294-1 Lecture #15

64

## 3c. Security and Resource Mgmt

- Q: Can untrusted users program the network?
- Need to isolate services and protect network
- A: Partly. This is difficult!
- Requires Protection
    - My program can't corrupt your program → Solved
  - Requires Resource Management
    - My program can't starve yours → Not solved

December 6, 2000

CS 294-1 Lecture #15

65

## Handling of Resource Mgmt Tasks

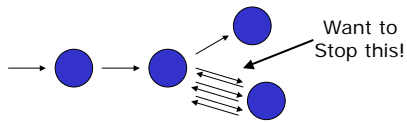
|                        | Internet                   | ANTS                     |
|------------------------|----------------------------|--------------------------|
| Per Packet At A Router | Limited by design of IP    | Limited for simple model |
| Per Packet Along Path  | Limited by protocol design | Whoops ... hole here     |
| Per User               | Need FQ or penalty boxes   | Same as Internet         |

December 6, 2000

CS 294-1 Lecture #15

66

## Example of Buggy Multicast



- How should we prevent this?
  - TTLs are a weak solution; not related to topology
  - Fairness mechanisms mitigate, but not enough
  - ANTS falls back on certification of programs ...

December 6, 2000

CS 294-1 Lecture #15

67

## Conclusions

- There are important applications
- Performance is not the limiting factor
- Protection is not the limiting factor
- Resource Management remains problematic

December 6, 2000

CS 294-1 Lecture #15

68

## Perspective

- The goal of this course was to give you perspective on networking/system design
- We've explored several axes:
  - The stack from physical media (IR, RF) to dynamic, adaptive applications
  - Wireless communication from wireless networking to wireless telephony
  - Moving from fixed to mobile: power, computation, ...

December 6, 2000

CS 294-1 Lecture #15

69

## Avenues for Further Exploration

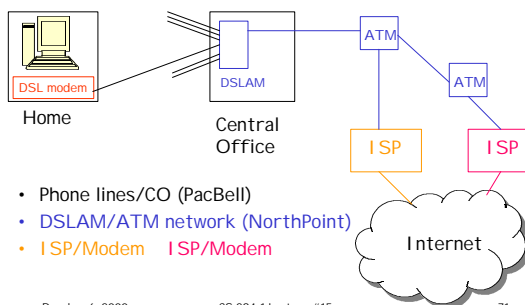
- After the last course offering, we started the Iceberg project:
  - Composable telecommunication services
  - Dynamic, robust wide-area soft-state protocols
  - Paths and APC for transcoding on the fly
- After this course offering...
  - Decomposing everything!
  - Separate ownership and management of all components

December 6, 2000

CS 294-1 Lecture #15

70

## Example Decomposed System:DSL



- Phone lines/CO (PacBell)
- DSLAM/ATM network (NorthPoint)
- ISP/Modem ISP/Modem

December 6, 2000

CS 294-1 Lecture #15

71

## Issues

- Dynamic composition of resources
  - Adaptive interfaces
- Dynamic (re)allocation of resources
  - Distributed monitoring
  - Resource exchange for pricing/purchase
- Billing, operations and maintenance
  - Lots of finger pointing!
  - Settlement processing

December 6, 2000

CS 294-1 Lecture #15

72

## Approaches

- Cross-layer integration and optimization
  - How to do this across providers?
- Dynamic service migration across providers
  - To optimize for ...
  - How to choose?
- Others?

December 6, 2000

CS 294-1 Lecture #15

73